Correspondence

Efficient Expressions for Completely and Partly Unsuccessful Batched Search of Tree-Structured Files

S. D. Lang and Yannis Manolopoulos

Abstract—**In this correspondence, closed-form, nonrecurrent expressions for the cost of completely and partly unsuccessful batched searching are developed for complete *j*-ary tree files. These expressions are applied to both the replacement and nonreplacement models of the search queries. The expressions provide more efficient formulas than previously reported for calculating the cost of batched searching. The expressions can also be used to estimate the number of block accesses for hierarchical file structures.**

Index Terms—**Access strategy, batched searching, performance evaluation, physical database design, sequential and tree-structured files.**

I. Introduction

In modeling the performance of a database system, it is important to accurately estimate the number of block accesses for satisfying a given query. This estimation is used in the physical design of database systems [17]; it is also used in the selection of access paths for query optimization, such as that used in the distributed database system SDD-1 [1], System R [15], and System R* [9]. Several results exist in the literature for estimating the number of block accesses for retrieving k records, in a file of T blocks each containing N/T records, assuming the k records are uniformly distributed among the T blocks of the file [2], [3], [19]. These results differ in their assumptions on whether the records are chosen with replacement (i.e., allowing duplications) or without replacement (i.e., not allowing duplications). Recently, these results have been extended to consider finite buffer storage and nonunique records for each key value, where exact and approximate formulas are obtained and validated on the R* distributed relational database system [10].

A problem closely related to estimating block accesses is known as batched searching [16]. In an off-line or batch environment, disk I/O can be greatly reduced if searches are processed in a batched fashion, compared to being processed individually. Shneiderman and Goodman argued for the many benefits of batched searching, and provided expressions for the expected cost (i.e., number of block accesses) of batched searching for sequential and tree-structured files, assuming the search queries satisfy a uniform distribution with replacement [16].

Recently, several results on batched searching have been reported. Palvia [12] obtained expressions for the cost of batched searching using a uniform distribution without replacement for the search queries. Also, Piwowarski [13] rounded out the work in [16] and obtained a closed-form expression for the batched searching cost of the tree-structured files considered in [16]. Recognizing the similarity between the analysis of batched searching and the analysis of estimating block accesses in database systems, Lang *et al.* [8] provided a unified analysis and obtained closed-form, nonrecurrent expressions for the expected cost of batched searching of sequential and tree-structured files. Extending the work in [12], Manolopoulos and Kollias [11] considered the case of search queries that are either completely or partly nonexisting in the file, and obtained expressions for the batched searching cost of sequential and tree-structured files, using both replacement and nonreplacement models.

One type of the tree-structured files being considered is the complete j-ary tree file. This is a multiway, balanced search tree where each node is completely filled with $j-1$ records (hence each node, except the leaf node, has j children). For such tree-structured files, the expressions of the batched searching cost obtained in [11] were computed by iterative and recurrent formulas. For example, let $\mathrm{COST}(k_1, k_2, l+1)$ denote the cost of batched searching k_1 existing records and k_2 nonexisting records in a complete j-ary tree with $l+1$ levels (using replacement or nonreplacement model). In [11], this expression is calculated as $1 + j\sum_{i=1}^{k_1}\sum_{m=1}^{k_2}\mathrm{COST}(i,m,l)\mathrm{PROB}(i,m,l)$, where $\mathrm{PROB}(i,m,l)$ gives the probability that i records out of the k_1 existing records and m records out of the k_2 nonexisting records in the batch, involve access to a subtree of l levels. Because this formula is iterative and recurrent, it requires an exponential number $\mathrm{O}((k_1k_2)^l)$ iterations to calculate the exact value. In each iteration, the expression $\mathrm{PROB}(i,m,l)$ requires calculating 6 binomial coefficients [11]. Faster but approximate formulas for calculating $\mathrm{COST}(k_1, k_2, l+1)$ were then developed requiring $\mathrm{O}(l)$ iterations; the relative error of the approximation was reported to be approximately less than 5% for a ternary tree file of 4 levels [11].

In this correspondence, we improve the efficiency of the formulas in [11] by extending the formulas developed in [8] to include search queries that are either completely or partly nonexisting in the complete j-ary file. We obtain *exact* and *efficient* expressions for the expected cost of batched searching. These expressions can also be used to estimate the number of block accesses for hierarchical file structures. Our expressions are iterative but nonrecurrent and require a *linear time* $\mathrm{O}(l)$ iterations in calculation. In addition, in each iteration only 4 binomial coefficients (not 6) need to be calculated. The derivation of these expressions is described in Section II. Section III concludes the paper and points out some future work.

II. Batched Searching of Tree Structured Files

Consider a complete j-ary tree with height l, where the root is at level 0 and the leaves are at level l. In the tree, there are j^i nodes at level i, $0 \le i \le l$. Suppose there are k search queries in a batch. The following theorem gives a closed-form expression for the expected number of node accesses in batched searching.

Theorem 1: For each node p at level i, $1 \le p \le j^i$, $0 \le i \le l$, let $Y_{i,p}$ denote the following random variable:

$$Y_{i,p} = \begin{cases} 1 & \text{if node } p \text{ at level } i \text{ is accessed} \\ & \text{in batched searching,} \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$\begin{aligned} \mathrm{COST}(k, l+1) &= \text{the expected number of node accesses} \\ &\quad \text{of batched searching} \\ &= \sum_{i=0}^{l}\sum_{p=1}^{j^i}\mathrm{PROB}[Y_{i,p}=1] \\ &= \sum_{i=0}^{l}\sum_{p=1}^{j^i}(1-\mathrm{PROB}[Y_{i,p}=0]). \end{aligned}$$

Manuscript received November 27, 1989; revised April 30, 1990. Recommended by M. Jarke.

S. D. Lang is with the Department of Computer Science, University of Central Florida, Orlando, FL 32816.

Y. Manolopoulos is with the Department of Electrical Engineering, Aristotelian University of Thessaloniki, Thessaloniki 54006, Greece.

IEEE Log Number 9039339.

0098-5589/90/1200-1433$01.00 © 1990 IEEE

Proof: This theorem has been proved in [8]. For completeness the proof is reproduced here. By definition,

$$\begin{aligned}\mathrm{COST}(k, l+1) &= \mathrm{E}\left[\sum_{i=0}^{l}\sum_{p=1}^{j^i} Y_{i,p}\right] \\ &= \sum_{i=0}^{l}\sum_{p=1}^{j^i} \mathrm{E}[Y_{i,p}] \\ &= \sum_{i=0}^{l}\sum_{p=1}^{j^i} \mathrm{PROB}[Y_{i,p}=1] \\ &= \sum_{i=0}^{l}\sum_{p=1}^{j^i} (1-\mathrm{PROB}[Y_{i,p}=0]).\end{aligned}$$

We now apply the theorem to the analysis of batched searching of uniformly distributed search keys. When k_1 keys are existing and k_2 keys are nonexisting in the file, there are four types of uniform distribution to consider, depending on whether the search keys follow a replacement or nonreplacement model [11]. The following theorem applies Theorem 1 to these cases and obtains iterative but nonrecurrent formulas for the expected cost of batched searching.

Theorem 2: Let $\mathrm{COST}(k_1, k_2, l+1)$ denote the expected number of node accesses in batched searching of $k_1 + k_2$ uniformly distributed keys, where k_1 search keys are existing, and k_2 keys are nonexisting in a complete j-ary tree of height l. Then $\mathrm{COST}(k_1, k_2, l+1)$ can be calculated as follows:

1) If the k_1 existing keys satisfy a replacement model and the k_2 nonexisting keys also satisfy a replacement model, then

$$\mathrm{COST}(k_1, k_2, l+1) = 1 + \sum_{i=1}^{l} j^i \left(1 - \frac{C(j^{l+1} - j^{l-i+1} + k_1 - 1, k_1)C(j^{l+1} - j^{l-i+1} + k_2 - 1, k_2)}{C(j^{l+1} + k_1 - 2, k_1)C(j^{l+1} + k_2 - 1, k_2)}\right).$$

2) If the k_1 existing keys satisfy a replacement model and the k_2 nonexisting keys satisfy a nonreplacement model, then

$$\mathrm{COST}(k_1, k_2, l+1) = 1 + \sum_{i=1}^{l} j^i \left(1 - \frac{C(j^{l+1} - j^{l-i+1} + k_1 - 1, k_1)C(j^{l+1} - j^{l-i+1}, k_2)}{C(j^{l+1} + k_1 - 2, k_1)C(j^{l+1}, k_2)}\right).$$

3) If the k_1 existing keys satisfy a nonreplacement model and the k_2 nonexisting keys satisfy a replacement model, then

$$\mathrm{COST}(k_1, k_2, l+1) = 1 + \sum_{i=1}^{l} j^i \left(1 - \frac{C(j^{l+1} - j^{l-i+1}, k_1)C(j^{l+1} - j^{l-i+1} + k_2 - 1, k_2)}{C(j^{l+1} - 1, k_1)C(j^{l+1} + k_2 - 1, k_2)}\right).$$

4) If the k_1 existing keys satisfy a nonreplacement model and the k_2 nonexisting keys also satisfy a nonreplacement model, then

$$\mathrm{COST}(k_1, k_2, l+1) = 1 + \sum_{i=1}^{l} j^i \left(1 - \frac{C(j^{l+1} - j^{l-i+1}, k_1)C(j^{l+1} - j^{l-i+1}, k_2)}{C(j^{l+1} - 1, k_1)C(j^{l+1}, k_2)}\right).$$

Proof: Note that in a complete j-ary tree of height l, there are a total of $j^{l+1} - 1$ existing keys and a total of j^{l+1} subintervals for nonexisting keys. Similarly, in any subtree at level i, $1 \le i \le l$, there are $j^{l-i+1} - 1$ existing keys and j^{l-i+1} subintervals for nonexisting keys. Since the proofs for these four cases are very similar, let us consider only case 3, where k_1 existing keys are uniformly distributed among $j^{l+1} - 1$ keys without replacement and k_2 nonexisting keys are uniformly distributed among j^{l+1} subintervals with replacement.

In this case, there are $C(j^{l+1} - 1, k_1)$ possible combinations for the k_1 existing keys, and there are $C(j^{l+1} + k_2 - 1, k_2)$ combinations for the k_2 nonexisting keys (see [6] for an explanation on these formulas for replacement and nonreplacement models). Therefore, there are $C(j^{l+1} - 1, k_1)C(j^{l+1} + k_2 - 1, k_2)$ many combinations of the $k_1 + k_2$ keys in the batch. Using a similar argument, for each subtree at level i, $1 \le i \le l$, there are $(j^{l+1} - 1) - (j^{l-i+1} - 1) = j^{l+1} - j^{l-i+1}$ keys existing in the nodes outside of the subtree, and there are $j^{l+1} - j^{l-i+1}$ subintervals outside of the subtree for nonexisting keys. Therefore, there are $C(j^{l+1} - j^{l-i+1}, k_1)C(j^{l+1} - j^{l-i+1} + k_2 - 1, k_2)$ combinations of $k_1 + k_2$ keys that do not involve access to the root node of the subtree in batched searching. Thus, using the notation in Theorem 1,

$$\begin{aligned}&\mathrm{PROB}[Y_{i,p}=0] \\ &\quad = \text{probability that node } p \text{ at level } i \\ &\qquad \text{is not accessed in batched searching} \\ &\quad = \begin{cases} \dfrac{C(j^{l+1}-j^{l-i+1},k_1)C(j^{l+1}-j^{l-i+1}+k_2-1,k_2)}{C(j^{l+1}-1,k_1)C(j^{l+1}+k_2-1,k_2)} & \text{if } i \ge 1, \\ 0 & \text{if } i = 0 \\ & \text{(the root is} \\ & \text{always accessed).} \end{cases}\end{aligned}$$

Thus, by applying Theorem 1,

$$\begin{aligned}&\mathrm{COST}(k_1, k_2, l+1) \\ &= 1 + \sum_{i=1}^{l}\sum_{p=1}^{j^i} \left(1 - \frac{C(j^{l+1} - j^{l-i+1}, k_1)C(j^{l+1} - j^{l-i+1} + k_2 - 1, k_2)}{C(j^{l+1} - 1, k_1)C(j^{l+1} + k_2 - 1, k_2)}\right) \\ &= 1 + \sum_{i=1}^{l} j^i \left(1 - \frac{C(j^{l+1} - j^{l-i+1}, k_1)C(j^{l+1} - j^{l-i+1} + k_2 - 1, k_2)}{C(j^{l+1} - 1, k_1)C(j^{l+1} + k_2 - 1, k_2)}\right).\end{aligned}$$

Note that in Theorem 2, if $k_2 = 0$, this is the case that all search keys are existing in the file and this case has been analyzed in [8]. If $k_2 \neq 0$, Theorem 2 analyzes the case of completely unsuccessful batched searching ($k_1 = 0$) and the case of partly unsuccessful batched searching ($k_1 \neq 0$), both have been considered in [11]. However, the results of Theorem 2 provide nonrecurrent formulas that are much more efficient for calculating the exact value of the batched searching cost.

III. Conclusion and Future Work

In this correspondence, we obtained closed-form, nonrecurrent expressions for the cost of batched searching for complete j-ary tree files. The search queries in the batch can be completely or partly nonexisting in the file, and the search queries are assumed to be uniformly distributed satisfying either a replacement or nonreplacement model. Our expressions provided more efficient formulas than previously reported for calculating the cost of batched searching. These expressions can also be used to estimate the number of block accesses for hierarchical file structures.

When large files are stored in a tree structure, it is possible that the data records are stored on the leaf level only (e.g., the B+ tree [5]), thus the nonleaf nodes are used as an index. In that case, expressions for the cost of batched searching have been obtained if the search queries are existing in the file [8]. As future work, we plan to apply Theorem 1 to such tree structures to include the case of search queries that are completely or partly nonexisting in the file.

Also, it should be noted that the results of Theorem 1 can be applied to search queries satisfying arbitrary distributions. It is well known that the assumption of uniform distribution may lead

to pessimistic estimation on block accesses in database systems, when the search distribution is in fact nonuniform [4]. Recently, Vander Zanden *et al.* [18] investigated approximation methods to estimate block accesses under nonuniform distributions. As future work, we plan to apply our results to consider batched searching with nonuniform distributions. In another direction, we plan to extend our work to cover batch insertions [7] and batch (incremental) updates [14] in tree-structured files.

ACKNOWLEDGMENT

The authors would like to thank the referees and editors for their speedy reviews and helpful suggestions.

REFERENCES

[1] P. A. Bernstein, N. Goodman, E. Wong, C. L. Reeve, and J. B. Rothnie, Jr., "Query processing in a system for distributed databases (SDD-1)," *ACM Trans. Database Syst.*, vol. 6, no. 4, pp. 602–625, Dec. 1981.
[2] A. F. Cardenas, "Analysis and performance of inverted database structures," *Commun. ACM*, vol. 18, no. 5, pp. 253–263, May 1975.
[3] T.-Y. Cheung, "Estimating block accesses and number of records in file management," *Commun. ACM*, vol. 25, no. 7, pp. 484–487, July 1982.
[4] S. Christodoulakis, "Implications of certain assumptions in database performance evaluation," *ACM Trans. Database Syst.*, vol. 9, no. 2, pp. 163–186, June 1984.
[5] D. Comer, "The ubiquitous B-tree," *ACM Comput. Surveys*, vol. 11, no. 2, pp. 121–138, June 1979.
[6] W. Feller, *An Introduction to Probability Theory and its Applications; Vol. I*, 2nd ed. New York: Wiley, 1957, pp. 28–32.
[7] S. D. Lang, J. R. Driscoll, and J. H. Jou, "Batch insertion for tree structured file organizations—Improving differential database representation," *Inform. Syst.*, vol. 11, no. 2, pp. 167–175, June 1986.
[8] ——,"A unified analysis of batched searching of sequential and tree-structured files," *ACM Trans. Database Syst.*, vol. 14, no. 4, pp. 604–618, Dec. 1989.
[9] G. M. Lohman, C. Mohan, L. M. Haas, B. G. Lindsay, P. G. Selinger, P. F. Wilms, and D. Daniels, "Query processing in R*," in *Query Processing in Database Systems*, Kim, Batory, and Reiner, Eds. New York: Springer-Verlag, 1985, pp. 31–47.
[10] L. F. Mackert and G. M. Lohman, "Index scans using a finite LRU buffer: A validated I/O model," *ACM Trans. Database System.*, vol. 14, no. 3, pp. 401–424, Sept. 1989.
[11] Y. Manolopoulos and J. G. Kollias, "Expressions for completely and partly unsuccessful batched search of sequential and tree-structured files," *IEEE Trans. Software Eng.*, vol. 15, no. 6, pp. 794–799, June 1989.
[12] P. Palvia, "Expressions for batched searching of sequential and hierarchical files," *ACM Trans. Database Syst.*, vol. 10, no. 1, pp. 97–106, Mar. 1985.
[13] M. Piwowarski, "Comments on batched searching of sequential and tree-structured files," *ACM Trans. Database Syst.*, vol. 10, no. 2, pp. 285–287, June 1985.
[14] N. Roussopoulos and H. Kang, "Principles and techniques in the design of ADMS±," *Computer*, vol. 19, no. 12, pp. 19–25, Dec. 1986.
[15] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price, "Access path selection in a relational database management system," in *Proc. ACM-SIGMOD Conf.*, 1979, pp. 23–34.
[16] B. Shneiderman and V. Goodman, "Batched searching of sequential and tree structured files," *ACM Trans. Database Syst.*, vol. 1, no. 3, pp. 268–275, Sept. 1976.
[17] T. J. Teorey and J. P. Fry, *Design of Database Structures*. Englewood Cliffs, N.J.: Prentice-Hall, 1982.
[18] B. T. Vander Zanden, H. M. Taylor, and D. Bitton, "A general framework for computing block accesses," *Inform. Syst.*, vol. 12, no. 2, pp. 177–190, June 1987.
[19] S. B. Yao, "Approximating block accesses in database organizations," *Commun. ACM*, vol. 20, no. 4, pp. 260–261, Apr. 1977.

Comments on "Measurement of Ada Overhead in OSI-Style Communications Systems"

Gerald M. Karam

***Abstract*—In "Measurement of Ada Overhead in OSI-Style Communications," Howes and Weaver compare the overhead between their proposed dispatcher-server architecture and Buhr's transport task architecture for OSI software implementation. This correspondence shows how their comparison method is weak because they did not consider: 1) the impact of control flow within a protocol, 2) the coordination required between multiple entities executing within a layer and which share the services of a lower layer, and 3) optimizations of Buhr's architecture which would have improved its performance efficiency. As a result the usefulness of their conclusions is very limited.**

***Index Terms*—Ada, communications software, concurrent systems, multiprocessor, open system interconnection (OSI), protocols.**

I. INTRODUCTION

The premise of the work of Howes and Weaver in [4] is that the Ada tasking structure proposed by Buhr [1] for OSI communications systems is inefficient because of the large number of tasks involved. The experiment they use to argue their case is the transmission of an integer value through the tasking structure of a seven layer OSI-like test system (see Fig. 1 in [4]). The transmission involves the addition of message headers when descending through the layers (during a message send) and the removal of the headers when ascending to the topmost layer (during a message receive). The outgoing and incoming message traffic are treated as independent activities. As an alternative, Howes and Weaver propose a dispatcher-server solution that allocates parallel transmitter servers on demand, with each server handling the operations of all seven layers (see [4, Fig. 3]). They do not consider data reception. Not surprisingly, their structure produces better experimental results because only two rendezvous are needed to send a message versus about $2N$ rendezvous for an N-layer implementation using Buhr's approach.

There are several significant failings in their comparison method: 1) their design does not support most protocols because it does not address control information exchange between communicating entities (Buhr's solution does), 2) their design does not support the resource sharing that occurs between the multiple entities within a layer or in a higher layer using the services provided by a lower level entity (Buhr's solution does), 3) Buhr's solution is examined in its most general form, thus, there are several optimizations which could reduce the rendezvous overhead costs so that Buhr's solution would become more attractive.

The discussion of these issues will begin with a brief tutorial on communication protocols and OSI, continue with a description of the problems in the comparison method, highlight a research effort that uses a more rigorous comparison method, and then conclude with a summary of the main points.

II. A BRIEF TUTORIAL ON PROTOCOLS AND OSI

A. Protocol Basics

A communications protocol is essentially an agreement between communicating entities on: 1) the format of messages, 2) the semantics of message contents, and 3) the legitimate sequence (or order) of messages. The message format includes fields containing coded information, identifiers etc., and may include data to be transmitted.

Manuscript received December 12, 1989; revised June 25, 1990. Recommended by T. Murata.

The author is with the Department of Systems and Computer Engineering, Carleton University, Ottawa, Ont. K1S 5B6, Canada.

IEEE Log Number 9039340.

0098–5589/90/1200–1435$01.00 © 1990 IEEE